Data Analysis (Prof. Ugo Ala) - R Practical 1

Fontanilla Natasha, Maiolino Aurelio, Mercadante Marianna

22-10-2025

Index

Introduction

Environment

The exrcise was run in a Docker, at the moment only aversion with JupyterLab is available. The image can be pulled from GitHub:

```
docker pull ghcr.io/maiolino-au/data_analysis:latest
```

To run it open a terminal in the working directory and run this:

- In Windows

```
@echo off
set "CURRENT_DIR=%cd%"
docker run -it --rm -p 8787:8787 -v "%CURRENT_DIR%:/sharedFolder"
 ↪  ghcr.io/maiolino-au/data_analysis:latest
```

- In Linux / MacOS

```
docker run -it --rm -p 8787:8787 -v .:/sharedFolder
 ↪  ghcr.io/maiolino-au/data_analysis:latest
```

Repository link: https://github.com/Maiolino-Au/Data_Analysis

The datas used are availabe in the repository or on moodle.

The scripts assume that the working directory is the same where the data are stored.

Working directory and Packages

First of all, we need to set the correct working directory: all the commands assume that the datas are stored in the working directory. I worked in a docker container to which i have shared a directory from my PC, called `/sharedFolder` inside the docker. In the directory there is one specific for this lesson, called `Practical_1`. Therefore:

```
setwd("/sharedFolder/Practical_1/")
```

Then we need to load the packages we are going to use. I used `suppressPackageStartupMessages()` to avoid printing all the startup messages of each package.

```
suppressPackageStartupMessages({
    library(phyloseq)
    library(dplyr)
    library(tidyr)
    library(stringr)
    library(microbiome)
    library(microbial)
    library(vegan)
    library(usedist)
    library(ggplot2)
    library(nortest)
    library(car)
})
```

Before we start: Let's look at the files

The objective of this first practical lesson is to start a basic analysis regarding 16S data. We are not starting from the real raw data, we alrtedy have the matrix of abbundances, the metadata (the subdivision of the experiments according to different classes), and the taxonomy.

The first step is to have a look at the different files

In R we look at the `phylosec`, able to combine all of these different forms of information:

- OTUs
- Abbundances
- Taxonomy
- Metadata
- The phylogenetic tree, not alwais present

We do not have the tree but you can download the other three files from moodle or the GitHub repository. There are three files for this exercise:

1. `otu.csv`
2. `metadata.csv`
3. `taxonomy.csv`

The abbundance matrix (otu.csv)

The first the abbundance matrix, very simiilar to the expression matrix. In columns we have the different subjects/samples and in the rows the different OTUs: cluster of reads clustered together according to the similarity (>97%) of their sequences. We basicaclly have build this sovrastructure of reads, and for each of these we have what we call counts: the number of reads assignd to that specific OTU in one specific sample.

We need to consider that this kind of experiments are compositional: an certain number in a sample is not equivalent to the same number in another sample, because the total number of counts can be different. For this reason we will need to normalize the data.

```
otu <- read.table(file = "otu.csv", sep = ",", header = T, row.names = 1, check.names =
 ↪  FALSE)
head(otu)
```

```
##         DISEASE_1 DISEASE_2 DISEASE_3 DISEASE_4 HEALTHY_1 HEALTHY_2 HEALTHY_3
## OTU101          0         0         0         0         0         0        44
## OTU1011        11       103         0        44       252       167       106
## OTU105          0         0         0         0         0         0        39
## OTU1052         0         0         0         0        32         0         0
## OTU106          0         0         0         0         0         0         0
## OTU1060         4         0         0        15        17         5        17
##         HEALTHY_4 BLANK_1 BLANK_2 BLANK_3 BLANK_4
## OTU101          0       0       0       0       0
## OTU1011       173     223      88      96      31
## OTU105          0       0       0       0       0
## OTU1052         0       0       0       0     121
## OTU106         12       0       8       5       0
## OTU1060         0      39      43      35      47
```

Metadata (metadata.csv)

The second file is the metadata, which contains the name of the samples and different categorizations:

- We have the status, which distinguishes the samples in Disease, Healty and Blank. Remember, we use blancks in this kind of experiments to have a control of possible contaminations: in this kind of studies the main problem is the possibility to have a lot of contaminants and the blancks help us to identify them.
- Another categorization is the Condition: blank or no_blank. The purpouse is to see wether we are able to distinguish blanks from all the other samples, are we able to identify something just in the blank which is so dissimilar from the other ones so that we can use this dissimilarity to clean/prune the information from the real samples according to this contaimnation?

```
metadata <- read.table(file = "metadata.csv", sep = ",", header = T, row.names = 1)
metadata
```

```
##            Status Condition
## DISEASE_1 DISEASE  NO_BLANK
## DISEASE_2 DISEASE  NO_BLANK
## DISEASE_3 DISEASE  NO_BLANK
## DISEASE_4 DISEASE  NO_BLANK
## HEALTHY_1 HEALTHY  NO_BLANK
## HEALTHY_2 HEALTHY  NO_BLANK
## HEALTHY_3 HEALTHY  NO_BLANK
## HEALTHY_4 HEALTHY  NO_BLANK
## BLANK_1     BLANK     BLANK
## BLANK_2     BLANK     BLANK
## BLANK_3     BLANK     BLANK
## BLANK_4     BLANK     BLANK
```

Taxonomy (taxonomy.csv)

The third and last file is the taxonomy, which contains the association between a specific OTU and the taxonomical classification. The taxonomy is given as a string with different levels, from the kingdom to the species or even strain, separated by semicolons, for example: `"k__Bacteria; p__Acidobacteria; c__Acidobacteria-6; o__iii1-15; f__Unknown_family; g__Unknown_genus403; s__Unknown_species1"`.

```
taxonomy <- read.table(file = "taxonomy.csv", sep = ",", header = T, row.names = 1)
head(taxonomy) # output not shown due to formatting: Taxon string is too long to display
 ↪ properly
```

The Phyloseq Object

We have seen the three different files and now we need to combine them into a single object, the **Phyloseq object**. We need to consider all the information togethere to be able to perform the analysis. For example, the first curiosity that we can have is understanting which are the most abbundant species inside the different samples. But the information we have right now is not sufficient: we need a statistic. Without statistical analysis we cannot state anyting, we can only describe or observe the data.

Importing the data

The first real steps is to import the data into R. To import the OTUs we used the `read.table` function. This is able to read a generic table and allows you to specify the separator, the header and the row names. The same function was used to import the taxonomy and the metadata, this is a standard step and not specific for this kind of analysis.

After importing the data we can have a look at the first lines with the `head()` function.

```
otu <- read.table(
    file = "otu.csv", # the name and of the file
    sep = ",", # the separator, this is a file is a csv ("comma-separated values")
    header = T, # indicates the presence of a header so the first row is not data
    row.names = 1, # indicates that the first column contains the row names
    check.names = FALSE # avoid R to change the names of the columns
)
```

Then we isolate the raw names to check if what we have imported is the right file.

```
taxa_present <- row.names(otu)
head(taxa_present)
```

```
## [1] "OTU101"  "OTU1011" "OTU105"  "OTU1052" "OTU106"  "OTU1060"
```

Cleaning the data in prepraration for the Phyloseq Object

After we have imported all the three files we need to prepare them for the analysis.

The taxonomy is provided in the "Greengenes" format. Greengenes is one of the databases of 16S rRNA sequences, it provides taxonomical classification in a specific format: a single string with different levels separated by semicolons. it is usefull to separate them into different columns, one for each taxonomical level. We can use the `separate()` function from the `tidyr` package to do this.

```
# clean the taxonomy, Greengenes format
tax <- taxonomy %>%
    select(Taxon) %>%
    separate(Taxon, c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus",
 ↪  "Species"), "; ")
head(tax)
```

```
##             Kingdom            Phylum              Class             Order
## OTU101  k__Bacteria  p__Acidobacteria c__Acidobacteria-6        o__iii1-15
## OTU1011 k__Bacteria  p__Acidobacteria           c__DA052       o__Ellin6513
```

```
## OTU105  k__Bacteria p__Actinobacteria  c__Actinobacteria o__Actinomycetales
## OTU1052 k__Bacteria  p__Bacteroidetes     c__Bacteroidia   o__Bacteroidales
## OTU106  k__Bacteria  p__Bacteroidetes     c__Bacteroidia   o__Bacteroidales
## OTU1060 k__Bacteria  p__Bacteroidetes     c__Bacteroidia   o__Bacteroidales
##                         Family              Genus               Species
## OTU101      f__Unknown_family g__Unknown_genus403    s__Unknown_species1
## OTU1011     f__Unknown_family  g__Unknown_genus18    s__Unknown_species1
## OTU105  f__Corynebacteriaceae  g__Corynebacterium s__Unknown_species1242
## OTU1052     f__Unknown_family g__Unknown_genus425    s__Unknown_species1
## OTU106      f__Unknown_family g__Unknown_genus580    s__Unknown_species1
## OTU1060     f__Unknown_family g__Unknown_genus813    s__Unknown_species1
```

We can notice one of the classical problems of 16S: it is not so specific. we can notice various "Unkwnown" at different taxonomical levels. This is due to the fact that the 16S rRNA gene is not able to distinguish between very similar species, therefore we can have a good classification up to a certain level, but not more specific.

NB: `%>%` is the pipe operator from the `dplyr` package, which allows to chain multiple operations. This operator takes the output of the left-hand side and passes it as the first argument to the function on the right-hand side. Therefore `taxonomy %>% select(Taxon)` is equivalent to `select(taxonomy, Taxon)`.

We have then various cleaning steps to do:

- Remove the prefixes like "k___","p___", etc.

```
tax.clean <- data.frame(
    row.names = row.names(tax),
    Kingdom = str_replace(tax[, 1], "k__", ""),
    Phylum = str_replace(tax[, 2], "p__", ""),
    Class = str_replace(tax[, 3], "c__", ""),
    Order = str_replace(tax[, 4], "o__", ""),
    Family = str_replace(tax[, 5], "f__", ""),
    Genus = str_replace(tax[, 6], "g__", ""),
    Species = str_replace(tax[, 7], "s__", ""),
    stringsAsFactors = FALSE
)
tax.clean[is.na(tax.clean)] <- ""
tax.clean[tax.clean == "__"] <- ""
head(tax.clean)
```

```
##          Kingdom         Phylum           Class           Order
## OTU101  Bacteria  Acidobacteria Acidobacteria-6         iii1-15
## OTU1011 Bacteria  Acidobacteria           DA052       Ellin6513
## OTU105  Bacteria Actinobacteria  Actinobacteria Actinomycetales
## OTU1052 Bacteria  Bacteroidetes     Bacteroidia   Bacteroidales
## OTU106  Bacteria  Bacteroidetes     Bacteroidia   Bacteroidales
## OTU1060 Bacteria  Bacteroidetes     Bacteroidia   Bacteroidales
##                    Family            Genus             Species
## OTU101      Unknown_family Unknown_genus403    Unknown_species1
## OTU1011     Unknown_family  Unknown_genus18    Unknown_species1
## OTU105  Corynebacteriaceae  Corynebacterium Unknown_species1242
## OTU1052     Unknown_family Unknown_genus425    Unknown_species1
## OTU106      Unknown_family Unknown_genus580    Unknown_species1
## OTU1060     Unknown_family Unknown_genus813    Unknown_species1
```

- Fill in missing taxonomical levels with “Unclassified ”

```
for (i in 1:nrow(tax.clean)) {
    if (tax.clean[i, 7] != "") {
        tax.clean$Species[i] <- paste(tax.clean$Genus[i], tax.clean$Species[i], sep = "
 ↪  ")
    } else if (tax.clean[i, 2] == "") {
        kingdom <- paste("Unclassified", tax.clean[i, 1], sep = " ")
        tax.clean[i, 2:7] <- kingdom
    } else if (tax.clean[i, 3] == "") {
        phylum <- paste("Unclassified", tax.clean[i, 2], sep = " ")
        tax.clean[i, 3:7] <- phylum
    } else if (tax.clean[i, 4] == "") {
        class <- paste("Unclassified", tax.clean[i, 3], sep = " ")
        tax.clean[i, 4:7] <- class
    } else if (tax.clean[i, 5] == "") {
        order <- paste("Unclassified", tax.clean[i, 4], sep = " ")
        tax.clean[i, 5:7] <- order
    } else if (tax.clean[i, 6] == "") {
        family <- paste("Unclassified", tax.clean[i, 5], sep = " ")
        tax.clean[i, 6:7] <- family
    } else if (tax.clean[i, 7] == "") {
        tax.clean$Species[i] <- paste("Unclassified ", tax.clean$Genus[i], sep = " ")
    }
}
head(tax.clean)
```

```
##          Kingdom         Phylum           Class           Order
## OTU101  Bacteria  Acidobacteria Acidobacteria-6         iii1-15
## OTU1011 Bacteria  Acidobacteria           DA052       Ellin6513
## OTU105  Bacteria Actinobacteria  Actinobacteria Actinomycetales
## OTU1052 Bacteria  Bacteroidetes     Bacteroidia   Bacteroidales
## OTU106  Bacteria  Bacteroidetes     Bacteroidia   Bacteroidales
## OTU1060 Bacteria  Bacteroidetes     Bacteroidia   Bacteroidales
##                     Family            Genus                              Species
## OTU101      Unknown_family Unknown_genus403   Unknown_genus403 Unknown_species1
## OTU1011     Unknown_family  Unknown_genus18    Unknown_genus18 Unknown_species1
## OTU105  Corynebacteriaceae  Corynebacterium Corynebacterium Unknown_species1242
## OTU1052     Unknown_family Unknown_genus425   Unknown_genus425 Unknown_species1
## OTU106      Unknown_family Unknown_genus580   Unknown_genus580 Unknown_species1
## OTU1060     Unknown_family Unknown_genus813   Unknown_genus813 Unknown_species1
```

Now that we have cleaned the taxonomy we can transform the three object in a table format compatible with Phyloseq.

```
OTU <- otu_table(as.matrix(otu), taxa_are_rows = TRUE)
# head(OTU)
```

```
TAX <- tax_table(as.matrix(tax.clean))
# head(TAX)
```

```
SAMPLE <- sample_data(metadata)
# SAMPLE[, ]
```

All that remain is to compine the three into a single Phyloseq object.

Generating the Phyloseq Object

The Phylosec object is created with the `phyloseq()` function, which takes as arguments three elements:

- The OTU table
- The Taxonomy table
- The Sample metadata

```
# merge the data into a Phyloseq Object
ps <- phyloseq(OTU, TAX, SAMPLE)
ps
```

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 966 taxa and 12 samples ]
## sample_data() Sample Data:       [ 12 samples by 2 sample variables ]
## tax_table()   Taxonomy Table:    [ 966 taxa by 7 taxonomic ranks ]
```

When we print the content of the phylosec object we can observe varius infromation: we can see that the object is formed by three big elements, the OTU table, the Taxonomy table and the Sample metadata. We can also see reported the macroclassification indide the three tables.

- The OTU table contains 12 samples and is comprised of 966 different OTUSs.
- The metadata (Sample Data) tells us that we have 12 samples (it is important that this number be consistent with the one shonw in the OTUs) and 2 variables or "covariate" (chiamate così perche se costruissimo un modello statistico queste sarebbero le due variabili da utilizzare per detarminare l'output?, non si capisce bene)
- The Taxonomy table needs to be consistent with the OTU for the number of taxa, and we see that there are the same 966. It also shows 7 taxonomy ranks, these are the 7 to which we have access. From the most generic one, Kindom, which seem obvius considering we are working with bacteria, but we need to remember that we can erroneusly include Archea or possibly Fungi (less likely, they have a different reference gene, the DS, but they could still be included).

Therefore now we have this first image, we know what was imported into phylosec and how it is organized

Plots of Relative Abundance

The second step is to plot the relative abbundance that we have in our samples. This is not our last goal, but just a first glimpse into the data to see which are the most abbundant species and genera inside our samples.

Selection of samples and taxa

The first step is a selection of our samples according to these categories: we want to extract the normal samples and the blanks, for now we are avoiding the disease ones. And we can see how to make a selection according to the phylosec metadata.

We use the `subset_samples()` function from the `phyloseq` package to select the samples according to the `Status` in the metadata. We select only the samples with `Status` equal to "HEALTHY" or "BLANK", and exclude the "DISEASE" samples.

```
##  library - microbial
##  Selection by metadata
psbar1 <- subset_samples(ps, Status == "HEALTHY" | Status == "BLANK")
metabar1 <- data.frame(sample_data(psbar1))
metabar1
```

```
##            Status Condition
## HEALTHY_1 HEALTHY  NO_BLANK
## HEALTHY_2 HEALTHY  NO_BLANK
## HEALTHY_3 HEALTHY  NO_BLANK
## HEALTHY_4 HEALTHY  NO_BLANK
## BLANK_1     BLANK     BLANK
## BLANK_2     BLANK     BLANK
## BLANK_3     BLANK     BLANK
## BLANK_4     BLANK     BLANK
```

Then filter against the rows that are all equal to zero. Could it be possible that an entire row contains only zeros? It is very unlikely that an entire transcript is all zero. We build the matrix starting from what we find in the reads and if there is a transcript in the table it means that at least one read should have been found. So why are we doing this? Why are we excluding the rows with only zeros, something that should not be present? Remember we have subletted the data: it is possbile that an OTU is present, but only in the disease samples, while being complitely absent from healty and blank ones. This is the classical case of differential abbundance: when there is a condition characterized by some species totaly absent in other conditions. This would be optimal because, if we have disease with 5 species that are not present in healty or blank (meaning they aren't contaminants) it means we have found an optimal result, that we have found exactly what we need. Thee same works for gene expression: if you see a gene extremly overexpressed in a tumor but practically absent or normal in wt it means you have found something important.

But you have to remember, when you analyze the data, of the selection you have made. Dov'è che mi frega se non metto questo check? Ovviamente nella statistica… perché? per due motivi:

- se ho tutti 0 riesco a fare una statistica? NO -> il sistema si potrebbe piantare li
- qualora non si piantasse, io aumento tantissimo il numero di n test per cui devo correggere poi, quindi se io posso abbassare il numero di correzioni che devo fare dopo è meglio. Se posso semplificare l'analisi la rende più robusta. Ad esempio con Bonferroni io moltiplico il p value nominale che ho calcolato per il numero di test che ho fatto.

Il mio oggetto phyloseq aveva 966 TAXA, vuol dire che io devo moltiplicare il mio pvalue per 966 (per 1000 praticamente). 0,001 moltiplicato per mille ottengo 1 che non è un buon p value… ma se io scopro, dopo aver pulito per gli 0, che quelli presenti alla fine sono solo più 50, anziché moltiplicare per 1000 , moltiplico per 50… è questo a cui devo fare attenzione!

```
# to find taxa that are "really" present in the considered subsection of samples
dat <- psbar1@otu_table[!apply(psbar1@otu_table, 1, function(x) all(x == 0)), ]
real_taxa <- row.names(dat@.Data)
head(real_taxa)
```

```
## [1] "OTU101"  "OTU1011" "OTU105"  "OTU1052" "OTU106"  "OTU1060"
```

Quindi questa è la riga, puliamo per tutto quello che non è zero, ne sopravvivono alcune, cioè se vediamo con questa pulizia non è che cambia di tanto, sono arrivato a 801… comunque ce n'erano circa 160 che erano presenti solo nei disease, ma non nelle altre categorie.

```
psbar2_otu <- psbar1@otu_table[c(real_taxa), ]
psbar2_taxa <- psbar1@tax_table[c(real_taxa), ]
SAMPLE1 <- sample_data(metabar1)
psbar_obj <- phyloseq(psbar2_otu, psbar2_taxa, SAMPLE1)
psbar_obj
```

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 801 taxa and 8 samples ]
## sample_data() Sample Data:       [ 8 samples by 2 sample variables ]
## tax_table()   Taxonomy Table:    [ 801 taxa by 7 taxonomic ranks ]
```

Normalization

```
##  check for the function "normalize"
##        which method?
##  method = "relative" argument, calculates relative abundances, which are the
 ↪  proportions of each feature
##  (like OTUs or ASVs) within a sample.
##  This method converts raw counts into percentages by dividing each feature's count by
 ↪  the total count
```

Ora NORMALIZZIAMO. Ma cosa vuol dire normalizzare in questo caso? Abbiamo detto che di default i nostri numeri counts non sono quelli immediatamente utilizzabili, per il problema composizionale. Allora possiamo optare per una normalizzazione… in questo caso normalize è una funzione all'interno del pacchetto che consente di optare per diverse normalizzazione, quella di default è questa relative, che sostanzialmente è proprio la normalizzazione più basale in assoluto: prendo il numero di counts in uno specifico OTU e lo divido per quanti counts ha quel sample… quindi è proprio relativa al sample. Per esempio: 50 counts per OTU 1, ne ho 1000 counts su quel sample -> faccio 50/1000.

Ma a questo punto questa frazione diventa comunque comparabile, cioè magari non è la normalizzazione più specifica o più figa, però rende i valori comparabili tra loro.

```
##  for that sample
phy <- suppressMessages(suppressWarnings({
    normalize(psbar_obj)
}))
```

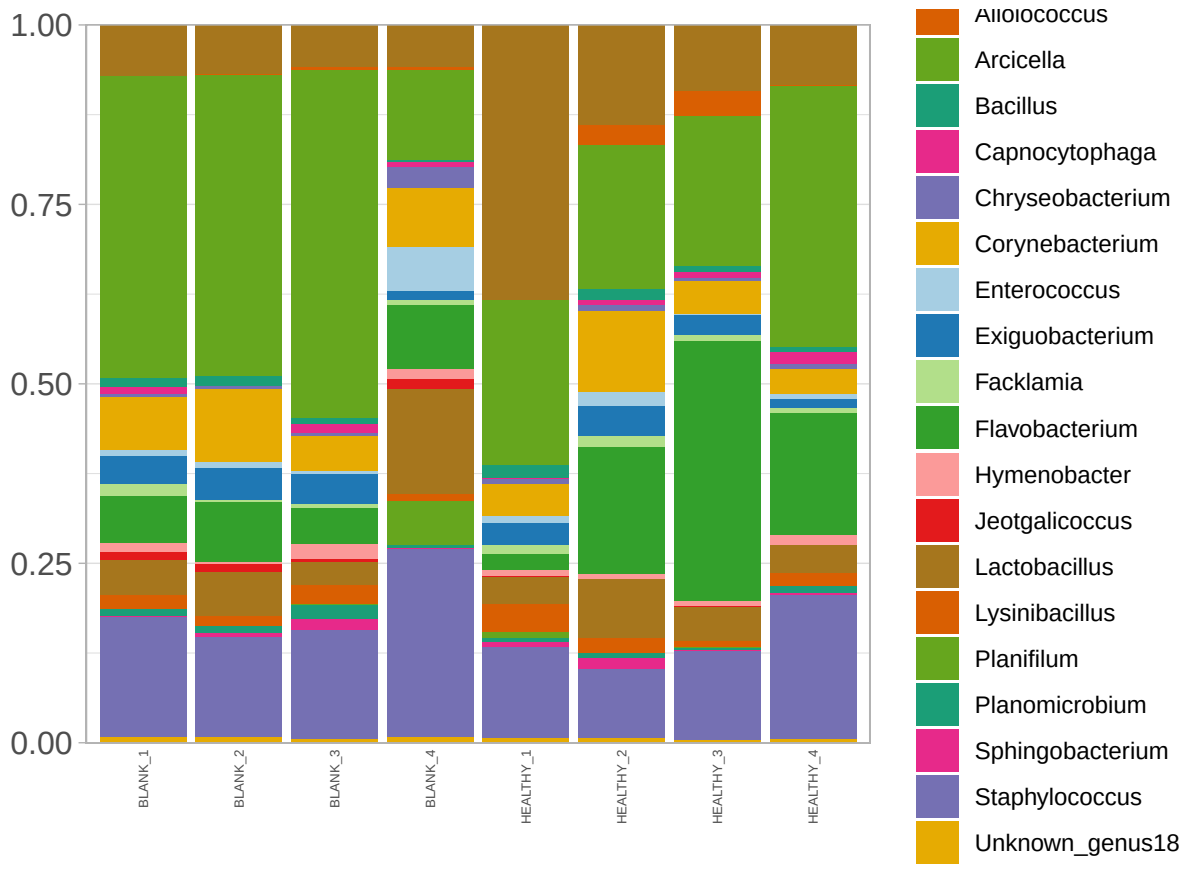
Quindi dopo la normalizzazione, dovremmo ottenere questo plot. Dove abbiamo:

- i nostri 4 + 4 samples (i 4 BLANK e i 4 HEALTHY)
- abbiamo chiesto a livello di Genus
- sono riportati i primi 20 generi

Questo ci dà un'idea di cosa ci sia nei nostri samples, e anche di come sono distribuiti.

Siamo contenti di questo però? Ci basta? No in realtà, però qui manca la statistica… se vediamo infatti il verde in alto (Arcicella) possiamo dire che c'è qualche segnale (c'è una differenza significativa tra i gruppi)? No.

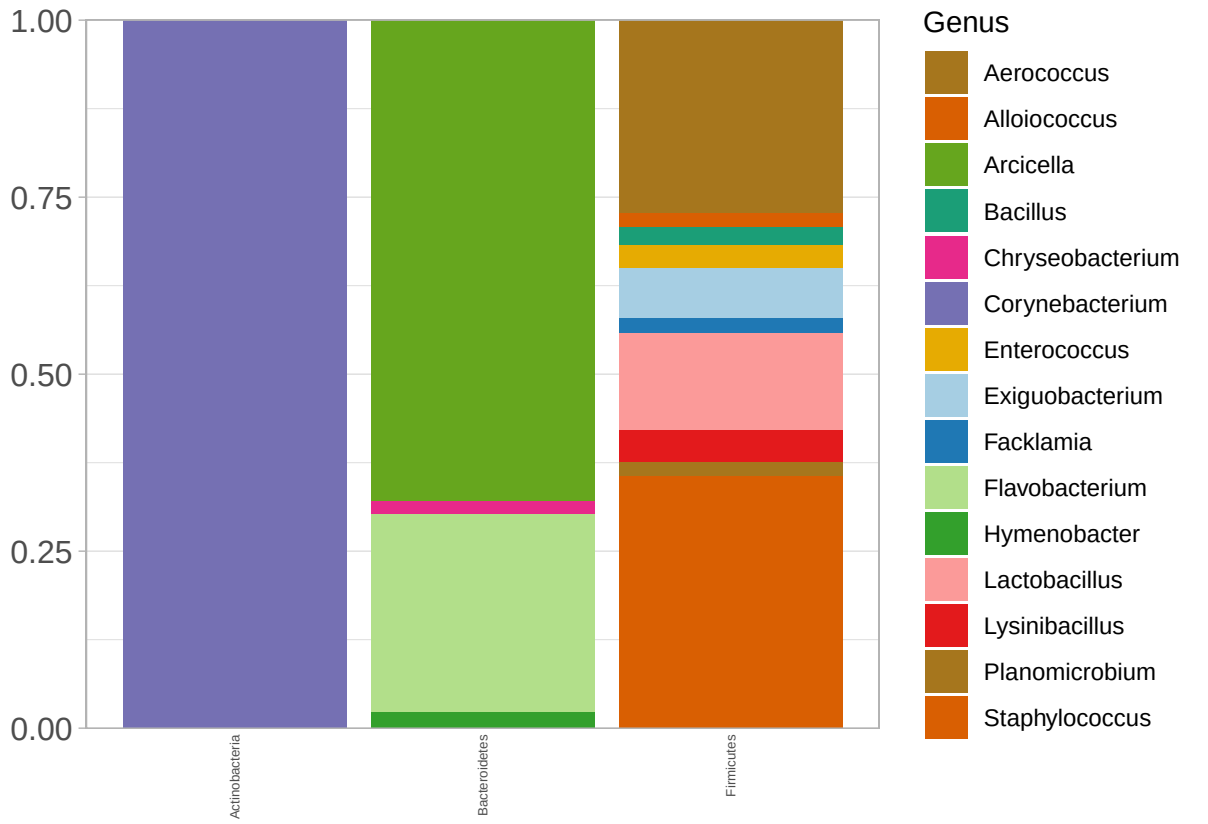
```
##  plot by samples
plotbar(phy, level = "Genus", top = 20)
```



Sarebbe carino trovassimo in un sample o una classe di sample, che presi i primi 15/10 il 90% è associato a uno specifico genere, perché allora quello ci sta dicendo che quel sample è veramente caratterizzato da quel genere.

Un'altra via di rappresentazione è quella di guardare sempre i generi, ma raggruppati per phylum e in questo caso cosa succede?

```
##  plot by taxonomical leyer
plotbar(phy, level = "Genus", top = 15, group = "Phylum")
```



Anziché avere i sample abbiamo i phyla diversi, quindi delle macro caratterizzazioni dei nostri batteri e all'interno di questi vediamo come sono suddivisi i nostri phyla.

Per esempio qui vediamo Actinobacteria (viola) che sostanzialmente è solo lui e non ha diversi generi al suo interno e poi ci sono i Bacteroidetes e i Firmicutes che sono un pochino più ricchi. Questo può esser più utile per vedere le classi maggiormente rappresentate e anche al loro interno come sono divise. Se dovessimo fare un' analogia con la gene expression (sulla quale abbiamo più espressione) quale potrebbe essere? Vedere dato un gene quali sono le isoforme presenti, quindi quelle generate da splicing alternativi, quindi non è il gene tp53 che mi interessa, ma quella isoforma specifica per andare a riconoscere le varie isoforme.

Recap

E' stato richiesto un RECAP logico su ciò che stiamo facendo in generale:

- Ho fatto dei prelievi di microbioma buccale (degli swap) per ognuno di noi
- Li ho mandati a sequenziare
- Ho fatto un esperimento di 16S
- Ho ottenuto dei FASTQ file: per ogni sample ho il mio raw data
- Di questi raw data sono andato a pulirli (quality control), il trimming, la ricostruzione (non ho fatto il percorso di denoising per arrivare agli ASV (Amplicon Sequence Variants), ma ho fatto una sorta di clustering per ottenere gli OTU (Operational Taxonomic Units).

- Ho messo insieme le reads che si assomigliano fino al 97% di confidenza.
- Ho costruito gli oggetti che vedevamo prima OTU11, OTU22
- Alla fine ottengo per ognuno di questi OTU che ho ricostruito (che per me sono semplicemente delle sequenze nucleotidiche adesso). Quindi in un OTU trovo una sequenza consenso accompagnata da tutte le sequenze che si avvicinano fino al 97% di similarità
- Quindi vedo se quella sequenza è presente in sample 1, in sample 2, 3 e quanto è rappresentata… cioè quante reads io posso associare a quella sequenza
- Questo mi genera questa tabella (otu.csv)

```
##         DISEASE_1 DISEASE_2 DISEASE_3 DISEASE_4 HEALTHY_1 HEALTHY_2 HEALTHY_3
## OTU101          0         0         0         0         0         0        44
## OTU1011        11       103         0        44       252       167       106
## OTU105          0         0         0         0         0         0        39
## OTU1052         0         0         0         0        32         0         0
## OTU106          0         0         0         0         0         0         0
## OTU1060         4         0         0        15        17         5        17
##         HEALTHY_4 BLANK_1 BLANK_2 BLANK_3 BLANK_4
## OTU101          0       0       0       0       0
## OTU1011       173     223      88      96      31
## OTU105          0       0       0       0       0
## OTU1052         0       0       0       0     121
## OTU106         12       0       8       5       0
## OTU1060         0      39      43      35      47
```

quindi vediamo che OTU1011 avrà 11 reads associate a disease 1, 103 associate a 103, non ne ha associate a disease 3, ne ha 44 associate a disease 4, ecc…

- Secondo step: ora abbiamo la sequenza consensus diciamo…come faccio a capire che cos'è? Noi sappiamo che deriva da 16S, quindi da una specifica regione (V3-V4) del 16s rRNA dei batteri però non so di quali…
- Allora devo fare la classificazione tassonomica che mi porta a questo oggetto qui:

```
##
## OTU101                k__Bacteria; p__Acidobacteria; c__Acidobacteria-6; o__iii1-15; f__Unknown_famil
## OTU1011                        k__Bacteria; p__Acidobacteria; c__DA052; o__Ellin6513; f__Unknown_fami
## OTU105  k__Bacteria; p__Actinobacteria; c__Actinobacteria; o__Actinomycetales; f__Corynebacteriaceae
## OTU1052             k__Bacteria; p__Bacteroidetes; c__Bacteroidia; o__Bacteroidales; f__Unknown_famil
## OTU106              k__Bacteria; p__Bacteroidetes; c__Bacteroidia; o__Bacteroidales; f__Unknown_famil
## OTU1060             k__Bacteria; p__Bacteroidetes; c__Bacteroidia; o__Bacteroidales; f__Unknown_famil
```

- L'OTU1011 per me è questo: a livello di genere è unknown, a livello di family è unknown, a livello di ordine iniziamo ad avere una prima classificazione

Facciamo un confronto con Blast… Lui prende la sequenza a cui sono interessato e la confronta con un database e come output ci dirà, se è un trascritto ci dirà il gene, se lo faccio con questo tipo di sequenza di 16S mi darà la specie. In che cosa questo tipo di approccio (quello che stiamo facendo noi) è più potente? Cioè richiede uno sviluppo non immediato. Non è scontato infatti quello che fa: qui ci sta dicendo ok la specie non la conosco, non conosco neanche il genere, non conosco neanche la famiglia, ma finalmente arrivando all'ordine e finalmente ho un riscontro.

Quindi cosa ha fatto questo classificatore? E' scontato dire non conosco né specie, né genere, né famiglia, ma ti so dire qualcosa sull'ordine..cosa ha dovuto fare?

Dovete immaginare che un genere ha sotto 100 specie, ma questo genere con altri 50 generi, fa parte di una famiglia, questa famiglia (che ha 50 generi e ognuno di questi ha 100 specie) appartiene con altre 70 famiglie a un ordine. Quindi vuol dire che all'interno di quell'ordine io ho potenzialmente centinaia se non migliaia di specie diverse, vuol dire che se lui me lo affibbia a quell'ordine, hai idea di quale pazzesca estrazione ha dovuto fare?

Blast questo non può farcelo, perchè lui ragiona sulla singola sequenza della singola specie, se la trova bene me la da (e me ne trova tantissime di solito), ma qui la potenzialità è che io risalgo all'interno dell'albero filogenetico e per ognuno di questi livelli ho il consensu di quello specifico gene. E' questa la potenzialità del nostro sistema.

Ovvio è che a ogni classificazione tassonomica è associata una confidenza, cioè c'è sempre uno score che va da 0 a 1… più è vicino a 1 e più sono contento. In alcuni casi queste associazioni sono con uno score dello 0,62, ma noi ci fidiamo di uno score 0,62? NO. Quindi per quanto sia potente e specifico in questa costruzione, non è detto che sia buona.

Ecco questo è un aspetto che è meno sentito nella gene expression, perchè di fatto quando riesco a dire che quello è il gene, il trascritto e anche l'isoforma specifica, difficilmente sto proprio sbagliando o quanto meno se una read è così brutta l'avrei già persa molto prima, quindi difficilmente arrivo qui.

Ma qui ho ancora un check che posso fare e posso tagliare e di tanto… perché io magari partivo da 1000 OTU caratterizzati, faccio la classificazione tassonomica e d aquella vado a tagliare quello che secondo me è troppo brutto e magari vado a tagliare del 50%. Quindi dai 1000 già solo per la classificazione tassonomica passo a 500. Quindi questo è un pochino quello che stavamo facendo…

Alpha and Beta Diversity

L'altro step importante è quello di fare una seconda caratterizzazione generale: l'**alpha diversity**. L'alpha diversity è sostanzialmente un numero fissato per ogni soggetto che dice qual è la complessità dal punto di vista dei TAXA presenti e di quanto sono abbondanti nei vari sample.

```
alpha <- suppressMessages(microbiome::alpha(ps, index = "all"))
head(alpha)
```

```
##           observed   chao1 diversity_inverse_simpson diversity_gini_simpson
## DISEASE_1      224 224.000                 30.034810             0.96670530
## DISEASE_2      237 237.125                  6.718725             0.85116224
## DISEASE_3       49  49.000                  1.027580             0.02683937
## DISEASE_4      237 237.000                  2.498256             0.59972076
## HEALTHY_1      258 258.000                  6.187914             0.83839464
## HEALTHY_2      289 289.200                 19.509541             0.94874303
##           diversity_shannon diversity_fisher diversity_coverage
## DISEASE_1         4.2314616        34.085270                 11
## DISEASE_2         3.0800671        34.016836                  3
## DISEASE_3         0.1173068         4.935205                  1
## DISEASE_4         2.1646418        32.173348                  1
## HEALTHY_1         3.0948937        36.559328                  2
## HEALTHY_2         4.1388815        42.665547                 10
##           evenness_camargo evenness_pielou evenness_simpson evenness_evar
## DISEASE_1        0.4946452      0.78191766       0.13408397     0.3078668
## DISEASE_2        0.4435090      0.56328332       0.02834905     0.3045524
## DISEASE_3        0.9979732      0.03014189       0.02097101     0.2476670
## DISEASE_4        0.8853422      0.39587016       0.01054116     0.3316111
## HEALTHY_1        0.7299785      0.55734129       0.02398416     0.3033624
## HEALTHY_2        0.5868802      0.73042179       0.06750706     0.3324804
##           evenness_bulla dominance_dbp dominance_dmn dominance_absolute
## DISEASE_1     0.43465458     0.1145371     0.1937593               2786
## DISEASE_2     0.27215858     0.3294698     0.4935670              11882
## DISEASE_3     0.01379723     0.9864844     0.9884208              99848
## DISEASE_4     0.21868967     0.6253711     0.7071627              31807
## HEALTHY_1     0.28721326     0.3387804     0.5422562              14367
## HEALTHY_2     0.39468993     0.1674940     0.2816164               6242
##           dominance_relative dominance_simpson dominance_core_abundance
## DISEASE_1          0.1145371        0.03329470               0.08325111
## DISEASE_2          0.3294698        0.14883776               0.44290705
## DISEASE_3          0.9864844        0.97316063               0.99032762
## DISEASE_4          0.6253711        0.40027924               0.21474214
## HEALTHY_1          0.3387804        0.16160536               0.75582437
## HEALTHY_2          0.1674940        0.05125697               0.49636408
##           dominance_gini rarity_log_modulo_skewness rarity_low_abundance
## DISEASE_1      0.9359701                   2.061396           0.11593488
## DISEASE_2      0.9643324                   2.061352           0.12638642
## DISEASE_3      0.9986290                   2.060839           0.01351565
## DISEASE_4      0.9736683                   2.061375           0.11694619
## HEALTHY_1      0.9593375                   2.061382           0.13714394
## HEALTHY_2      0.9265585                   2.061391           0.17345104
##           rarity_rare_abundance
## DISEASE_1           0.834320013
```

```
## DISEASE_2           0.502606477
## DISEASE_3           0.007301217
## DISEASE_4           0.745561432
## HEALTHY_1           0.164167138
## HEALTHY_2           0.328065044
```

Con quell' "all" noi diciamo ad R di calcolarci tutte le possibili alpha diversity, che se vedete non sono neanche poche. Gli observed sono il numero di OTU che riusciamo ad associare, vediamo che c'è chao1, simpson, shannon ecc.. Ma cosa vuol dire quel 224?

Vuol dire che il sample disease 1 è caratterizzato da 224 OTU che hanno un valore di abbondanza superiore a 1. Chao1 è molto simile ovvero aggiusta l'abbondanza del numero puro attraverso il conteggio di quanti singleton e doubleton ci sono, cioè quanti Otu con valori 1 o 2 ci sono. Perché il problema è sempre questo se io trovo un OTU presente con una sola read associata, mi posso fidare? Dipende...o è un errore clamoroso oppure ho proprio trovato una giusta osservazione.

Chi ce lo dice? In questo caso ci aiuta le varie confidenze.. la qualità della read che ho sequenziato, l'associazione tassonomica e il relativo score di confidenza. Quindi il chao1 spessissimo è uguale, ma per esempio nel disease 2 c'è almeno un OTU o due che hanno valore 1 di abbondanza e quindi fanno sì che questo aggiusti leggermente il valore di alpha diversity per questo sample.. mentre gli altri sono uguali se notate (224->224, 49->49, 237-> 237).

```
aa <- merge(alpha, metadata,
    by = "row.names", all = TRUE
)
# head(aa)
```

Ma una volta ottenute le nostre alpha diversity cosa dobbiamo fare?

Dobbiamo associare i **metadati**, ma prima di ciò il prof ha fatto una specie di giochetto in cui ha trasformato la tabella di abbondanza (ps) in forma composizionale.

```
# Use relative abundance data
ps1 <- microbiome::transform(ps, "compositional")

# Pick core taxa
ps1 <- core(
    ps1,
    detection = 0.001, # sets the minimum abundance threshold. 0 means that a taxon is
     ↪  considered "detected" even if it appears at any nonzero abundance.
    prevalence = 60 / 100 # sets the prevalence threshold, i.e. the minimum fraction of
     ↪  samples in which a taxon must be detected to be considered part of the core.
)
```

Chiediamo quindi di trasformare i nostri OTU con la loro abbondanza in counts come se fossero delle frazioni. Prima abbiamo usato "normalize" che era già una prima forma.

Qui quello che succede è che noi andiamo a struttura con una sorta di analisi delle componenti principali i nostri sample, ma con questa richiesta: abbiamo selezionato gli OTU che hanno

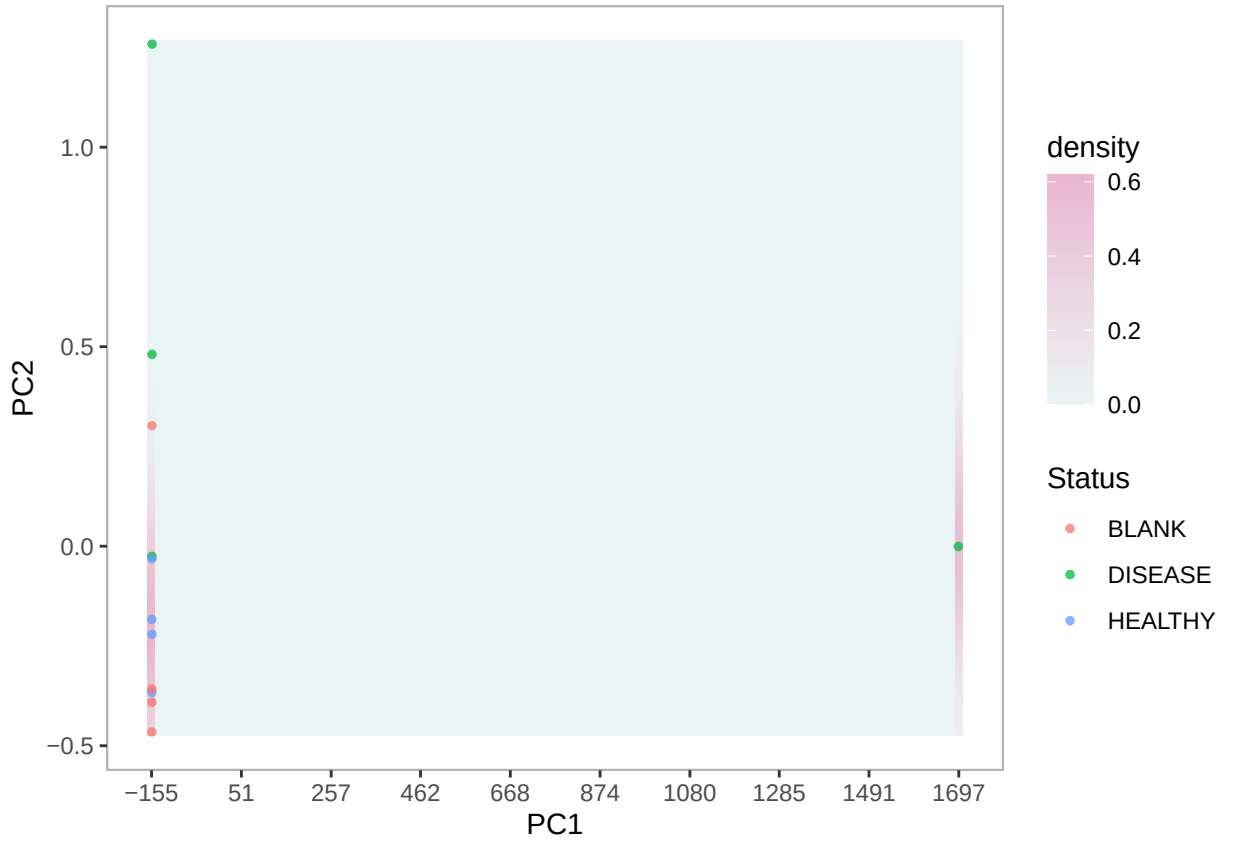
- detection = 0.001,
- prevalence = 60/100

Dove la detection è il livello di abbondanza relativa/composizionale che deve avere nel sample che io considero (sotto quello è come se non ci fosse) e poi la prevalence che è la percentuale di campioni in cui questo avviene.

Questo (ovvero la richiesta di una % di prevalenza con almeno un minimo di livello di abbondanza relativa) ci riporta verso a quel concetto che avevamo visto? Vi ricordate quali sono i due livelli del microbioma di cui parlavamo? C'è il **core** e il **transiente**. Con questa richiesta cerchiamo di isolare la porzione del microbioma più robusta, cioè quella presente in tutti i campioni. Sorge però una domanda importante: chi mi garantisce che c'è un core microbiota tra tutto inclusi i "blank"? Il concetto di "core microbiome" deve essere definito in modo operativo, a seconda del contesto sperimentale. Ad esempio, possiamo chiederci: esiste un "core microbiome" del soggetto sano? La risposta non è scontata. Dal Human Microbiome Project abbiamo imparato che dall'analisi di tanti tessuti diversi, il contesto del "sano" ha senso se il contesto è ben definito. Nel caso del microbiota intestinale, una persona sana può avere un core microbiome condiviso, ma fattori come stile di vita e dieta influenzano fortemente la composizione. C'è una componente di microbioma del host che interagisce con tutto il microbioma esterno ed è in grado di coevolvere. La coevoluzione indica che il microbioma cambia da persona a persona. Per questo motivo i metadati sono fondamentali: permettono di distinguere correttamente le classi di campioni e interpretare la variabilità biologica.

Noi qui abbiamo fatto una selezione di dati di questo tipo e siamo arrivati ad ottenere questo plot. Riusciamo a clusterare i samples in questo spazio virtuale multidimensionale. Come possiamo commentare questo risultato?

```
# Illustrate sample similarities with PCoA (NMDS)
plot_landscape(ps1, "NMDS", "bray", col = "Status")
```



Ce n'è un campione molto diverso, appartenente al gruppo DISEASE, mentre tutti gli altri risultano raggruppati altrove. Inoltre, ci sono altri DISEASE che non cadono come gli altri, vi è una sorta di finto cluster dove abbiamo gli HEALTHY e i BLANK e insieme a tre DISEASE. Non è un risultato brutto, colpisce perché in qualche modo i DISEASE hanno una variabilità molto alta, mentre gli altri sono tutti assieme li.

Dobbiamo pensare a come caratterizzare la variabilità interna che è un aspetto fondamentale e legata all'alpha

diversity. L'alpha diversity misura la ricchezza , l'abbondanza e l'eveness (ovvero come i nostri campioni si stanno comportando) delle specie presenti nei campioni. Ci aspettiamo che la variabilità interna dei DISEASE sia maggiore rispetto agli altri gruppi. Va inoltre considerato il **numero di samples**: in questo caso solo 4 per il gruppo DISEASE, quindi con un **potere statistico ridotto**.

Qua possiamo vedere la differenza tra matrice originaria (basata sui counts) e trasformata in matrice composizionale (abbondanze normalizzate OTU). Questa trasformazione serve per **normalizzare i dati** e consentire **confronti significativi** tra campioni.

HEALTHY_HAIR or HEALTHY_CSF and NO_BLANK

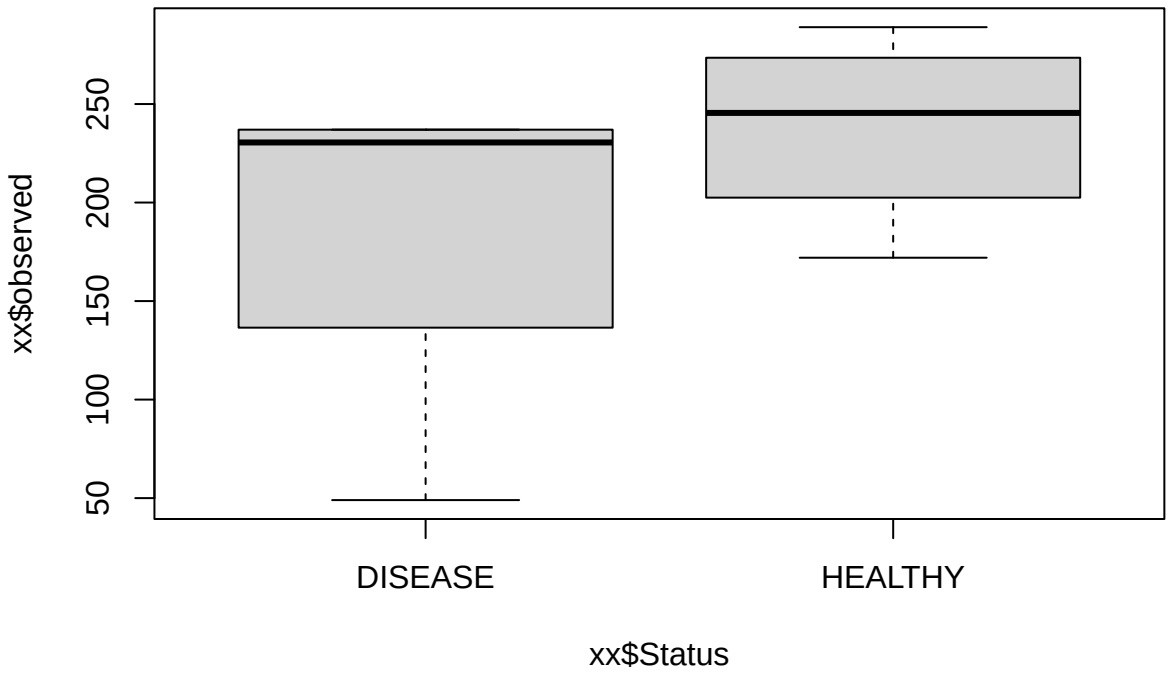
Ora inizia a subsettare/selezionare un sottoinsieme di samples con condizione NO_BLANK che sono i DISEASE e HEALTHY. Se andate a vedere il metadata, nel campo "Condition " indica se un campione è " BLANK" o "NO_BLANK". Scegliendo "NO_BLANK" vedete i DISEASE e HEALTHY, isolando i campioni di interesse.

```r
xx <- subset(aa, Condition == "NO_BLANK")
# head(xx)
# tail(xx)
```

Che test posso effettuare? Posso fare un WILCOXON test che consente di confrontare due gruppi indipendenti (HEALTHY vs DISEASE) e fornisce il p-value , evidenziando le differenza tra gruppi nell'alpha diversity (definita dagli "observed").

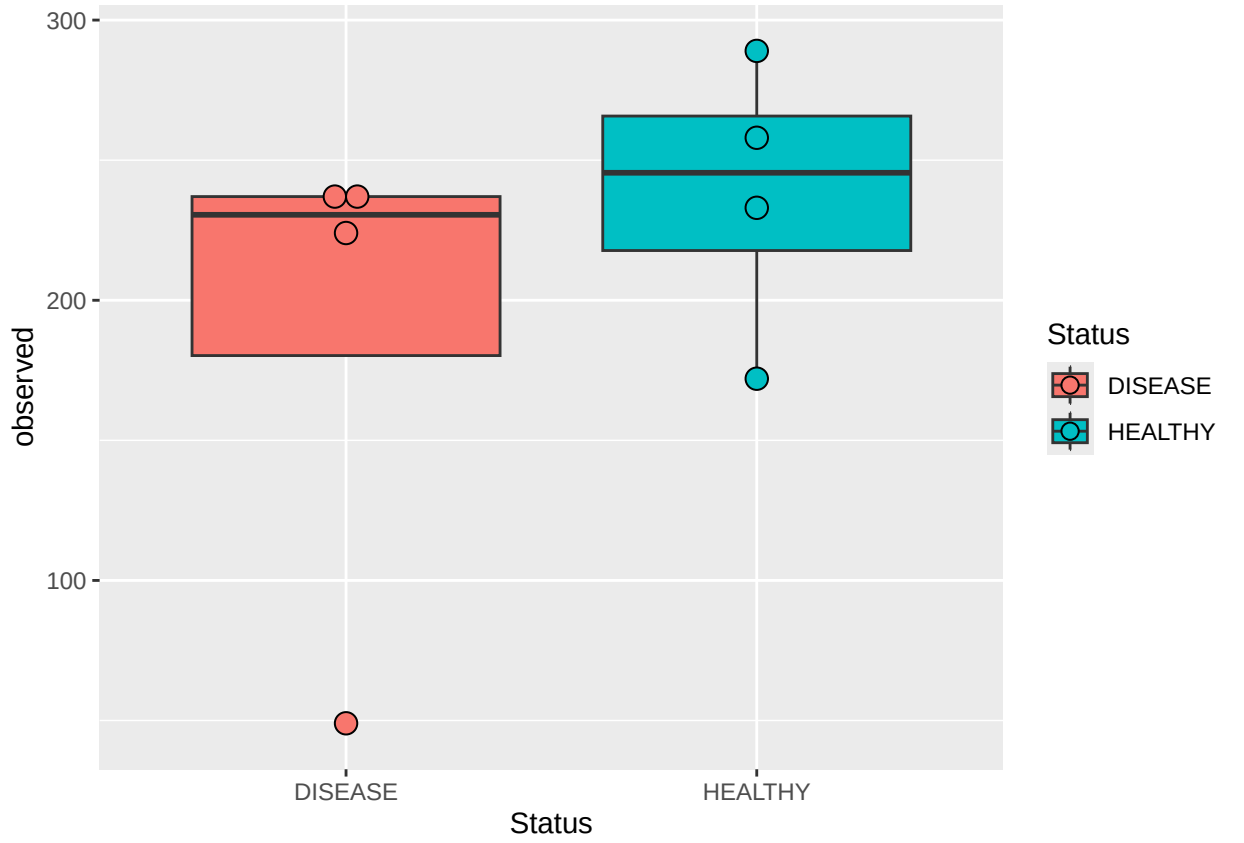
Nota: `suppressMessages(suppressWarnings({}))` è stato aggiunto poichè l'output delle cue funzioni conteneva caratteri non riconosiuti dal renderer di LaTex (usato da Rmarkdown). Nella repository su GitHub è presente un file di JupyterLab (`Practical_1/Steps.ipynb`) in cui sono visibili anche tutti gli output.

```r
### observed
suppressMessages(suppressWarnings({
    wilcox.test(xx$observed ~ xx$Status)
    boxplot(xx$observed ~ xx$Status)
}))
```



Il boxplot mostra la variabilità di cui parlavamo prima. Vediamo una dispersione dei DISEASED? A metà, in realtà nei DISEASED c'è una sorta di outlier che sembra governare tutto (in basso, isolato). Se perdessimo questo outlier DISEASED (in basso, isolato), vedete come sono compatti gli altri 3. A differenza, gli HEALTHY mantengono una loro variabilità intrinseca.

```
ggplot(xx, aes(x = Status, y = observed, fill = Status)) +
    geom_boxplot() +
    geom_dotplot(
        binaxis = "y", stackdir = "center",
        binwidth = diff(range(xx$observed, na.rm = TRUE)) / 30 # sets the width of the
        ↪  dots in the dotplot, put explitly for reasons regarding Rmarkdown
    )
```



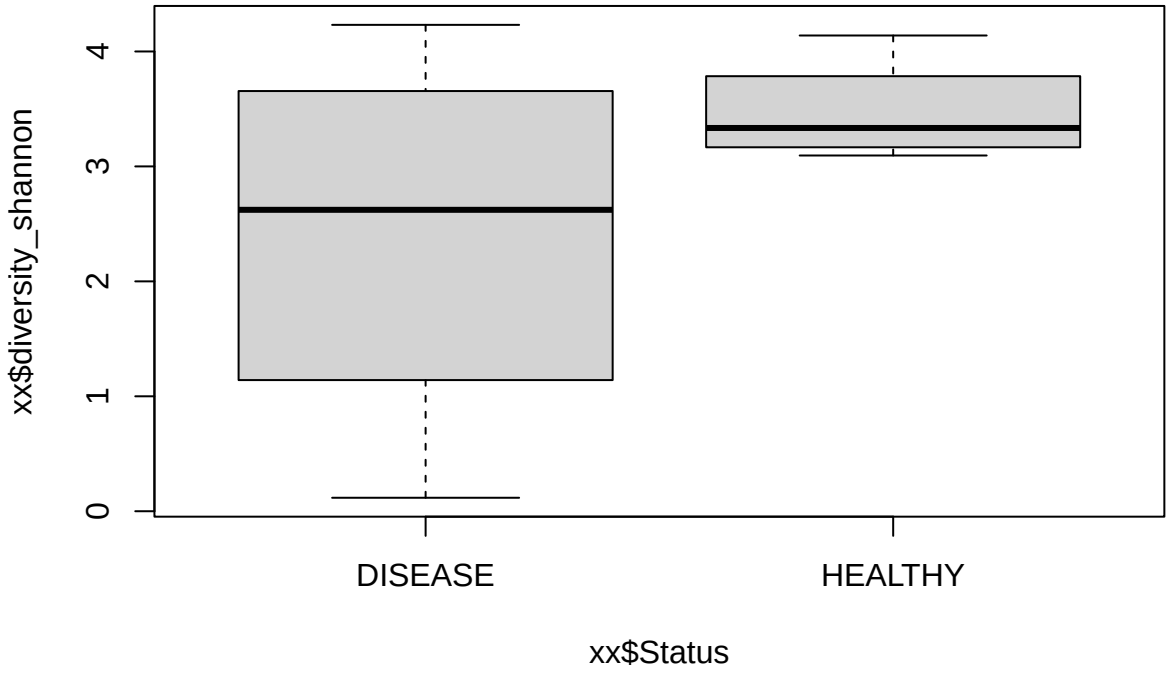
Se faccio la stessa cosa sullo Shannon guardo il p-value (0,34 che è lontano dalla significatività di 0,05), guardo la distribuzione dei pallini cambia un po'. La differenza tra "observed" e "Shannon" è che:

- Observed –> misura la richness (numero di OTU/specie presenti);
- Shannon –> considera anche la evenness, cioè la distribuzione relativa delle abbondanze.

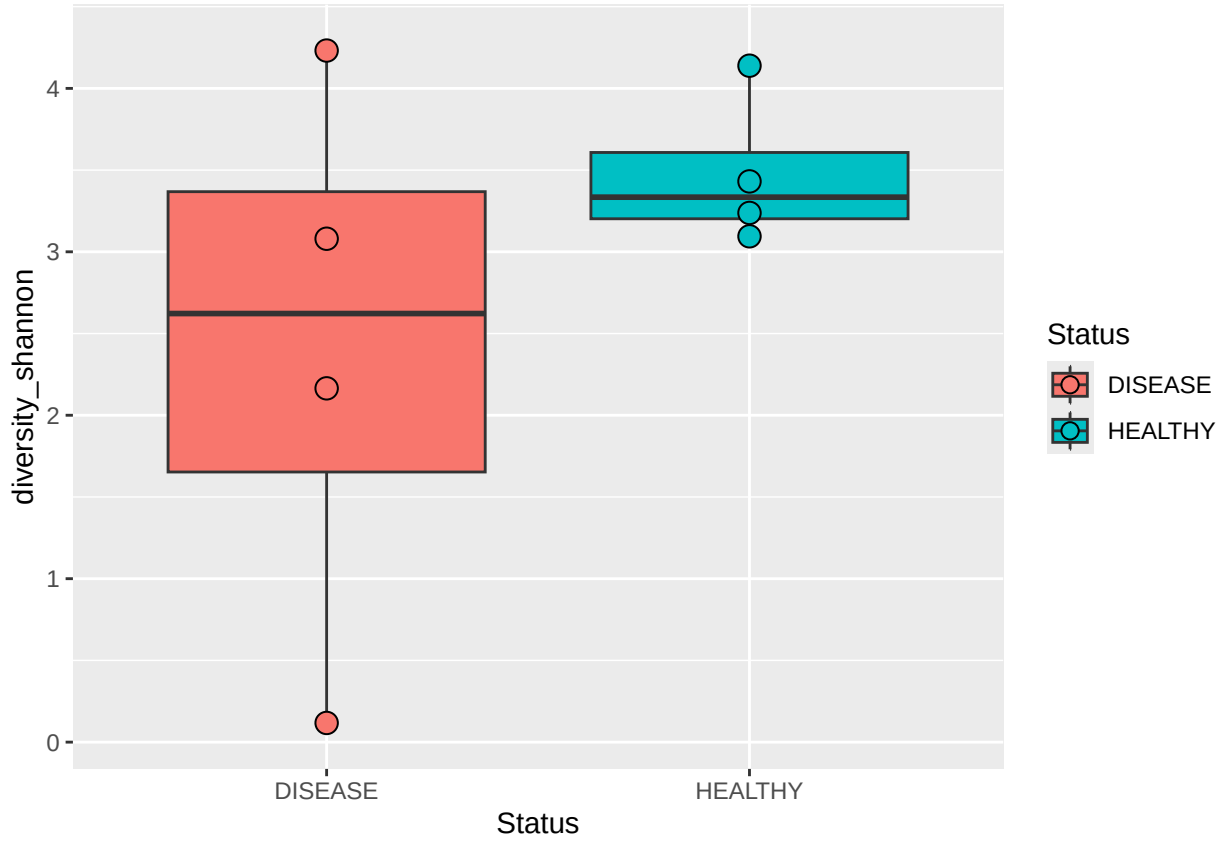
In questo modo vediamo che i HEALTHY sono molto più raggruppati : l'abbondanza relativa fa sì che ci sia meno dispersione nei normali.

```
### diversity_shannon
suppressMessages(suppressWarnings({
    wilcox.test(xx$diversity_shannon ~ xx$Status)
```

```
    boxplot(xx$diversity_shannon ~ xx$Status)
}))
```

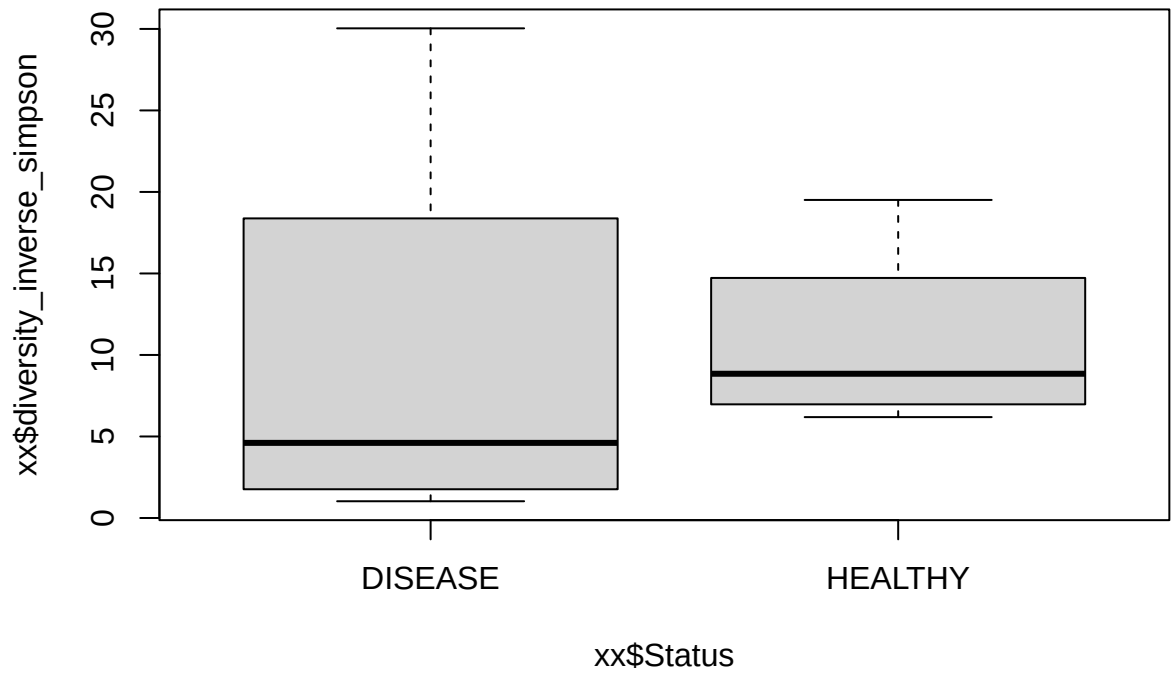



```
ggplot(xx, aes(x = Status, y = diversity_shannon, fill = Status)) +
    geom_boxplot() +
    geom_dotplot(
        binaxis = "y", stackdir = "center",
        binwidth = diff(range(xx$diversity_shannon, na.rm = TRUE)) / 30
    )
```

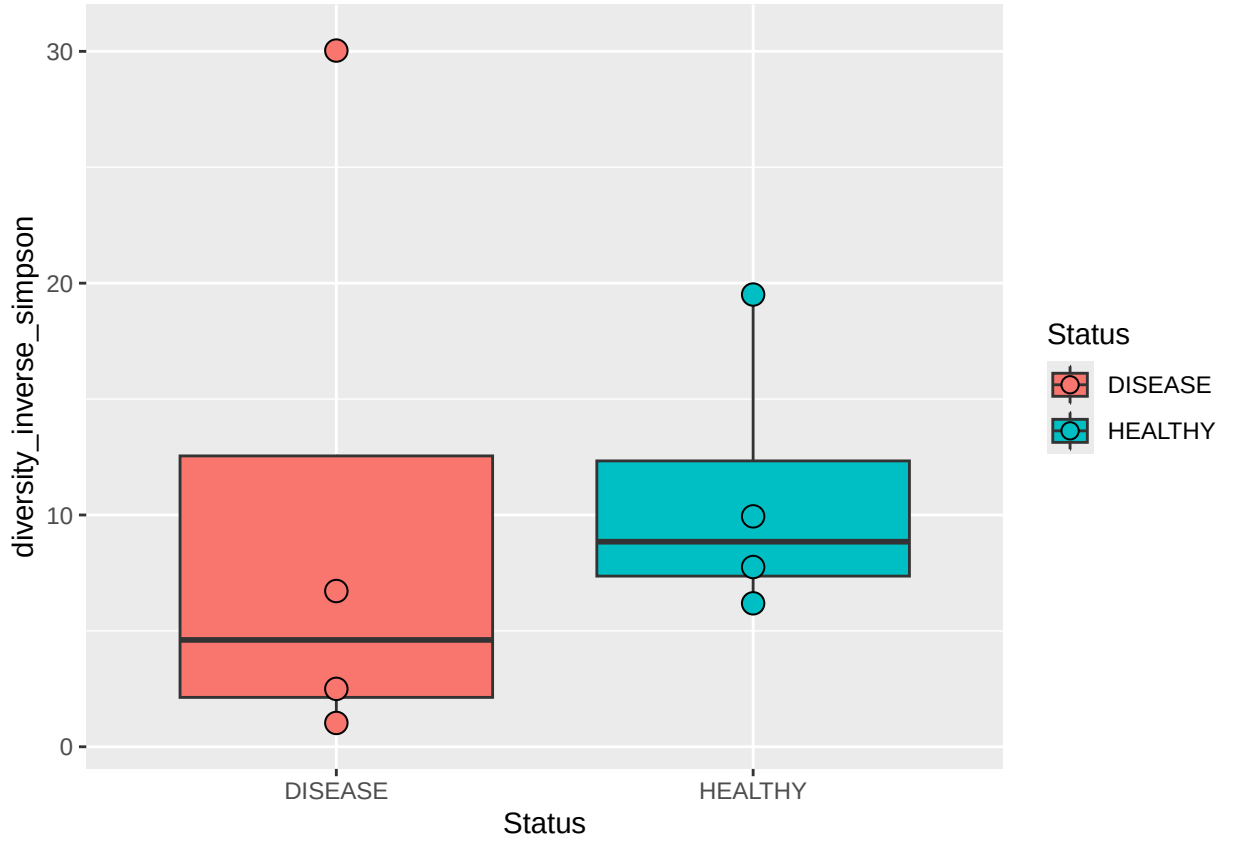


Se guardiamo SIMPSON, i risultati si avvicinano agli observed.

```
### diversity_inverse_simpson
suppressMessages(suppressWarnings({
    wilcox.test(xx$diversity_inverse_simpson ~ xx$Status)
    boxplot(xx$diversity_inverse_simpson ~ xx$Status)
}))
```




```
ggplot(xx, aes(x = Status, y = diversity_inverse_simpson, fill = Status)) +
    geom_boxplot() +
    geom_dotplot(
        binaxis = "y", stackdir = "center",
        binwidth = diff(range(xx$diversity_inverse_simpson, na.rm = TRUE)) / 30
    )
```



Dovessimo aggiungere un test da valutare, quale sceglieremmo? Prima, abbiamo considerato la richness e eveness. Un test possibile è basato sulla distanza filogenetica ma non possiamo farla perché non abbiamo il dato. Vi ricordate il plot che riporta le correlazioni tra i vari alpha? Potremmo sceglierne una o due in più che sono completamente scorrelate tra questi due per avere una prospettiva diversa.

Vorrei farvi notare che quando abbiamo calcola il primo WILCOXON sugli observed dice:

```
Warning message in wilcox.test.default(x = DATA[[1L]], DATA[[2L]], ...): "cannot compute
exact p-value with ties"
```

(Questo era uno dei messaggi che fanno fallire il render: ho dovuto riscriverlo manualmente, facendo copia incolla dall'output della funzione il render da errore)

Perchè dice questo? Il motivo per cui dice questo è perché si basa sul ranking e se ho tanti punti pari merito, occupano lo stesso rank.

Il messaggio "cannot compute exact p-value with ties" compare perché ci sono valori identici (ties) nel dataset. Il test di Wilcoxon si basa sui ranghi: se più valori sono uguali, condividono lo stesso rango, rendendo impossibile calcolare un p-value esatto.

Per esempio, se ho 5 corridori che arrivano nello stesso istante, allora ho 5 posti pari merito. Non so chi sia il vincitore e qui è la stessa cosa: devo prendere quale specie si trovano nella stessa posizione e ne tanti specie che hanno lo tesso valore, non so chi mettere prima o dopo nel ranking. Dobbiamo ricordarci di questa problematica.

Exercise

```
############################
###   question: run the statistics without excluding any sample
###   which is the test?
###   kruskal.test(A ~ B, data = data)
###   pairwise.wilcox.test(data$A, data$B, p.adjust.method = "BH")
############################
```

Qui avete un esercizietto.

Mi domando se le 3 condizioni DISEASE, HEALTHY and BLANK sono diversi tra di loro secondo gli observed, shannon e simpson? Uso la stessa strategia ma devo cambiare test. Non utilizzerò piu il Wilcoxon test che è su 2 gruppi, ma utilizzerò il Kruskal-Wallis test che considera da 3 gruppi in su. La problematica di avere 3 gruppi rispetto a 2 dal punto di vista delle analisi:

- Se ho 2 gruppi -> capisco in quale dei due gruppi c'è un segnale maggiore (es. alpha maggiore nei diseased).
- Se ho 3 gruppi -> ho i gruppi DISEASE, HEALTHY and BLANK e un p-value significativo di 0,01 e c'è una differenza in Shannon. Il test non ci dice quale è il gruppo/i che differiscono quindi dobbiamo fare questo pairwise comparison (seconda riga).

La pairwise comparison farà tutti i confronti a coppie possibili e mi dice quali sono quelli significativi. La pairwise Wilcoxon test farà :

- DISEASE vs HEALTHY
- DISEASE vs BLANK
- HEALTHY vs BLANK

Accade la stessa cosa con la gene expression TUMOR, METASTASIS, HEALTHY (3 gruppi). Se ho la gene expression su tre, il p-value che ottengo è globale indipendentemente dal test (DSeq2/ANOVA ecc) che uso. Per sapere la condizione o le condizioni importanti, si devono fare confronti a coppie.

La stessa cosa accade su quella composizionale. Non si parte dalla matrice con i counts, ma da quella trasformata per vedere se e come cambia.

```
############
### ALPHA on COMPOSITIONAL DATA
##########
ps_c <- microbiome::transform(ps, "compositional")
alpha_c <- microbiome::alpha(ps_c, index = "all")
```

```
## Observed richness
```

```
## Other forms of richness
```

```
## Diversity
```

```
## Warning in diversities_help(x, index = idx, zeroes = TRUE): Fisher diversity defined only for intege
##             the OTU table contains non-integers. Fisher not estimated.
```

```
## Evenness
```

```
## Dominance

## Rarity
```

```
head(alpha_c)
```

```
##           observed chao1 diversity_inverse_simpson diversity_gini_simpson
## DISEASE_1      224   224                 30.034810             0.96670530
## DISEASE_2      237   237                  6.718725             0.85116224
## DISEASE_3       49    49                  1.027580             0.02683937
## DISEASE_4      237   237                  2.498256             0.59972076
## HEALTHY_1      258   258                  6.187914             0.83839464
## HEALTHY_2      289   289                 19.509541             0.94874303
##           diversity_shannon diversity_coverage evenness_camargo evenness_pielou
## DISEASE_1         4.2314616                 11        0.4946452      0.78191766
## DISEASE_2         3.0800671                  3        0.4435090      0.56328332
## DISEASE_3         0.1173068                  1        0.9979732      0.03014189
## DISEASE_4         2.1646418                  1        0.8853422      0.39587016
## HEALTHY_1         3.0948937                  2        0.7299785      0.55734129
## HEALTHY_2         4.1388815                 10        0.5868802      0.73042179
##           evenness_simpson evenness_evar evenness_bulla dominance_dbp
## DISEASE_1       0.13408397     0.3078668     0.43465458     0.1145371
## DISEASE_2       0.02834905     0.3045524     0.27215858     0.3294698
## DISEASE_3       0.02097101     0.2476670     0.01379723     0.9864844
## DISEASE_4       0.01054116     0.3316111     0.21868967     0.6253711
## HEALTHY_1       0.02398416     0.3033624     0.28721326     0.3387804
## HEALTHY_2       0.06750706     0.3324804     0.39468993     0.1674940
##           dominance_dmn dominance_absolute dominance_relative dominance_simpson
## DISEASE_1     0.1937593          0.1145371          0.1145371        0.03329470
## DISEASE_2     0.4935670          0.3294698          0.3294698        0.14883776
## DISEASE_3     0.9884208          0.9864844          0.9864844        0.97316063
## DISEASE_4     0.7071627          0.6253711          0.6253711        0.40027924
## HEALTHY_1     0.5422562          0.3387804          0.3387804        0.16160536
## HEALTHY_2     0.2816164          0.1674940          0.1674940        0.05125697
##           dominance_core_abundance dominance_gini rarity_log_modulo_skewness
## DISEASE_1               0.08325111      0.9359701                    2.059840
## DISEASE_2               0.44290705      0.9643324                    2.060954
## DISEASE_3               0.99032762      0.9986290                    2.060839
## DISEASE_4               0.21474214      0.9736683                    2.061352
## HEALTHY_1               0.75582437      0.9593375                    2.060617
## HEALTHY_2               0.49636408      0.9265585                    2.060768
##           rarity_low_abundance rarity_rare_abundance
## DISEASE_1           0.11593488           0.834320013
## DISEASE_2           0.12638642           0.502606477
## DISEASE_3           0.01351565           0.007301217
## DISEASE_4           0.11694619           0.745561432
## HEALTHY_1           0.13714394           0.164167138
## HEALTHY_2           0.17345104           0.328065044
```

```
aa_c <- merge(alpha_c, metadata,
    by = "row.names", all = TRUE
)
head(aa_c)
```

```
##   Row.names observed chao1 diversity_inverse_simpson diversity_gini_simpson
## 1   BLANK_1      196   196                  6.507532              0.8463319
## 2   BLANK_2      183   183                  6.451069              0.8449869
## 3   BLANK_3      158   158                  5.396675              0.8147007
## 4   BLANK_4      278   278                 35.281113              0.9716562
## 5 DISEASE_1      224   224                 30.034810              0.9667053
## 6 DISEASE_2      237   237                  6.718725              0.8511622
##   diversity_shannon diversity_coverage evenness_camargo evenness_pielou
## 1          3.275834                 4        0.5394110       0.6206448
## 2          3.230167                 4        0.5534409       0.6200548
## 3          3.144660                 3        0.4929293       0.6211558
## 4          4.464470                13        0.7490272       0.7933139
## 5          4.231462                11        0.4946452       0.7819177
## 6          3.080067                 3        0.4435090       0.5632833
##   evenness_simpson evenness_evar evenness_bulla dominance_dbp dominance_dmn
## 1       0.03320169     0.2706940      0.3403487    0.37626048     0.4366116
## 2       0.03525174     0.2749752      0.3361593    0.37669368     0.4339581
## 3       0.03415617     0.3038682      0.3789414    0.41992994     0.4671277
## 4       0.12691048     0.3506440      0.4611997    0.09790011     0.1610005
## 5       0.13408397     0.3078668      0.4346546    0.11453708     0.1937593
## 6       0.02834905     0.3045524      0.2721586    0.32946983     0.4935670
##   dominance_absolute dominance_relative dominance_simpson
## 1         0.37626048         0.37626048        0.15366809
## 2         0.37669368         0.37669368        0.15501308
## 3         0.41992994         0.41992994        0.18529928
## 4         0.09790011         0.09790011        0.02834378
## 5         0.11453708         0.11453708        0.03329470
## 6         0.32946983         0.32946983        0.14883776
##   dominance_core_abundance dominance_gini rarity_log_modulo_skewness
## 1               0.67667963      0.9618133                   2.060652
## 2               0.68222925      0.9628243                   2.060621
## 3               0.67055865      0.9637944                   2.060631
## 4               0.34548562      0.9140507                   2.060658
## 5               0.08325111      0.9359701                   2.059840
## 6               0.44290705      0.9643324                   2.060954
##   rarity_low_abundance rarity_rare_abundance  Status Condition
## 1           0.09807435             0.1952679   BLANK     BLANK
## 2           0.09112245             0.2461433   BLANK     BLANK
## 3           0.07270523             0.2080115   BLANK     BLANK
## 4           0.14766578             0.5806427   BLANK     BLANK
## 5           0.11593488             0.8343200 DISEASE  NO_BLANK
## 6           0.12638642             0.5026065 DISEASE  NO_BLANK
```

```r
###########
### HEALTHY_HAIR or HEALTHY_CSF and NO_BLANK
##########
xx_c <- subset(aa_c, Condition == "NO_BLANK")
head(xx_c)
```

```
##    Row.names observed chao1 diversity_inverse_simpson diversity_gini_simpson
## 5  DISEASE_1      224   224                 30.034810              0.96670530
## 6  DISEASE_2      237   237                  6.718725              0.85116224
## 7  DISEASE_3       49    49                  1.027580              0.02683937
```

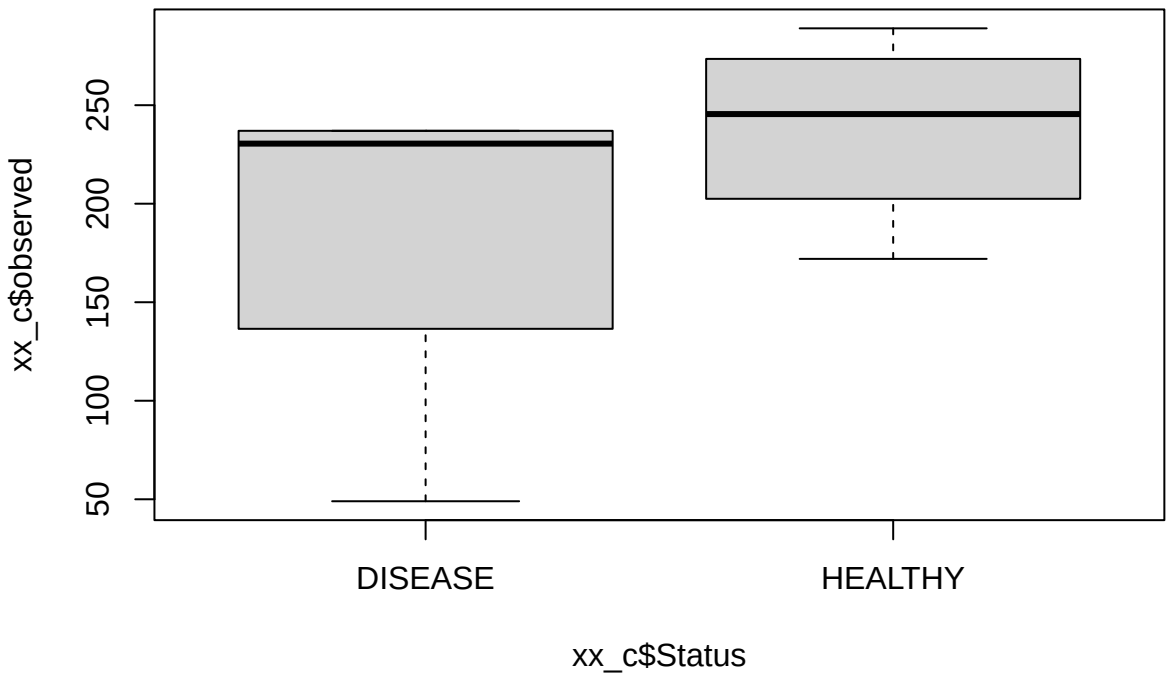
```
## 8  DISEASE_4      237   237                 2.498256            0.59972076
## 9  HEALTHY_1      258   258                 6.187914            0.83839464
## 10 HEALTHY_2      289   289                19.509541            0.94874303
##    diversity_shannon diversity_coverage evenness_camargo evenness_pielou
## 5          4.2314616                11        0.4946452      0.78191766
## 6          3.0800671                 3        0.4435090      0.56328332
## 7          0.1173068                 1        0.9979732      0.03014189
## 8          2.1646418                 1        0.8853422      0.39587016
## 9          3.0948937                 2        0.7299785      0.55734129
## 10         4.1388815                10        0.5868802      0.73042179
##    evenness_simpson evenness_evar evenness_bulla dominance_dbp dominance_dmn
## 5        0.13408397     0.3078668     0.43465458     0.1145371     0.1937593
## 6        0.02834905     0.3045524     0.27215858     0.3294698     0.4935670
## 7        0.02097101     0.2476670     0.01379723     0.9864844     0.9884208
## 8        0.01054116     0.3316111     0.21868967     0.6253711     0.7071627
## 9        0.02398416     0.3033624     0.28721326     0.3387804     0.5422562
## 10       0.06750706     0.3324804     0.39468993     0.1674940     0.2816164
##    dominance_absolute dominance_relative dominance_simpson
## 5           0.1145371          0.1145371        0.03329470
## 6           0.3294698          0.3294698        0.14883776
## 7           0.9864844          0.9864844        0.97316063
## 8           0.6253711          0.6253711        0.40027924
## 9           0.3387804          0.3387804        0.16160536
## 10          0.1674940          0.1674940        0.05125697
##    dominance_core_abundance dominance_gini rarity_log_modulo_skewness
## 5                0.08325111      0.9359701                   2.059840
## 6                0.44290705      0.9643324                   2.060954
## 7                0.99032762      0.9986290                   2.060839
## 8                0.21474214      0.9736683                   2.061352
## 9                0.75582437      0.9593375                   2.060617
## 10               0.49636408      0.9265585                   2.060768
##    rarity_low_abundance rarity_rare_abundance  Status Condition
## 5            0.11593488          0.834320013 DISEASE  NO_BLANK
## 6            0.12638642          0.502606477 DISEASE  NO_BLANK
## 7            0.01351565          0.007301217 DISEASE  NO_BLANK
## 8            0.11694619          0.745561432 DISEASE  NO_BLANK
## 9            0.13714394          0.164167138 HEALTHY  NO_BLANK
## 10           0.17345104          0.328065044 HEALTHY  NO_BLANK
```

```
tail(xx_c)
```

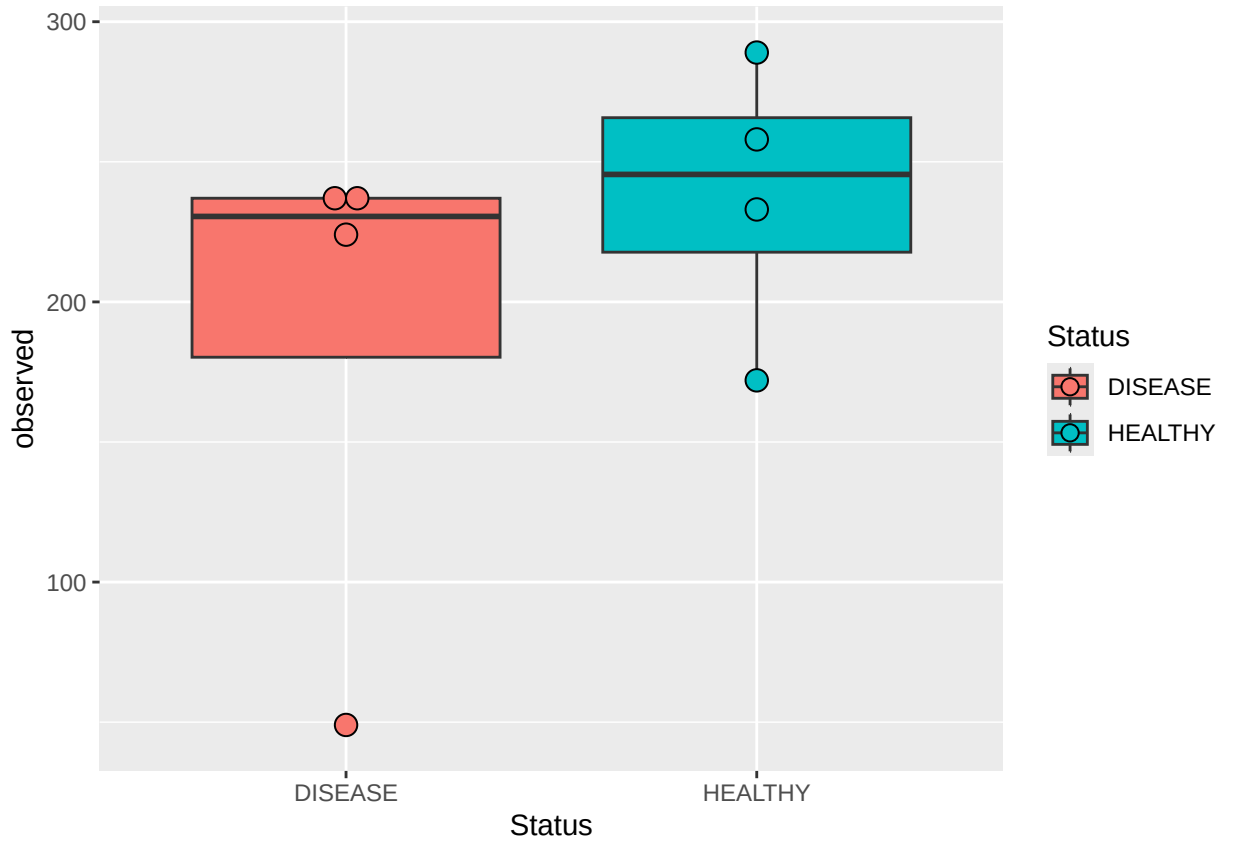
```
##    Row.names observed chao1 diversity_inverse_simpson diversity_gini_simpson
## 7  DISEASE_3       49    49                 1.027580            0.02683937
## 8  DISEASE_4      237   237                 2.498256            0.59972076
## 9  HEALTHY_1      258   258                 6.187914            0.83839464
## 10 HEALTHY_2      289   289                19.509541            0.94874303
## 11 HEALTHY_3      233   233                 9.939775            0.89939410
## 12 HEALTHY_4      172   172                 7.754238            0.87103826
##    diversity_shannon diversity_coverage evenness_camargo evenness_pielou
## 7          0.1173068                 1        0.9979732      0.03014189
## 8          2.1646418                 1        0.8853422      0.39587016
## 9          3.0948937                 2        0.7299785      0.55734129
## 10         4.1388815                10        0.5868802      0.73042179
```

```
## 11         3.4309010                 4        0.4775319      0.62940320
## 12         3.2379803                 3        0.6049407      0.62904006
##    evenness_simpson evenness_evar evenness_bulla dominance_dbp dominance_dmn
## 7        0.02097101     0.2476670     0.01379723     0.9864844     0.9884208
## 8        0.01054116     0.3316111     0.21868967     0.6253711     0.7071627
## 9        0.02398416     0.3033624     0.28721326     0.3387804     0.5422562
## 10       0.06750706     0.3324804     0.39468993     0.1674940     0.2816164
## 11       0.04265998     0.3078565     0.30643784     0.2353225     0.4169852
## 12       0.04508278     0.3041162     0.34340664     0.3212764     0.4314378
##    dominance_absolute dominance_relative dominance_simpson
## 7           0.9864844          0.9864844        0.97316063
## 8           0.6253711          0.6253711        0.40027924
## 9           0.3387804          0.3387804        0.16160536
## 10          0.1674940          0.1674940        0.05125697
## 11          0.2353225          0.2353225        0.10060590
## 12          0.3212764          0.3212764        0.12896174
##    dominance_core_abundance dominance_gini rarity_log_modulo_skewness
## 7                 0.9903276      0.9986290                   2.060839
## 8                 0.2147421      0.9736683                   2.061352
## 9                 0.7558244      0.9593375                   2.060617
## 10                0.4963641      0.9265585                   2.060768
## 11                0.6794495      0.9567662                   2.060402
## 12                0.7324300      0.9650276                   2.060713
##    rarity_low_abundance rarity_rare_abundance  Status Condition
## 7            0.01351565           0.007301217 DISEASE  NO_BLANK
## 8            0.11694619           0.745561432 DISEASE  NO_BLANK
## 9            0.13714394           0.164167138 HEALTHY  NO_BLANK
## 10           0.17345104           0.328065044 HEALTHY  NO_BLANK
## 11           0.14528590           0.224457768 HEALTHY  NO_BLANK
## 12           0.09457714           0.191078251 HEALTHY  NO_BLANK
```

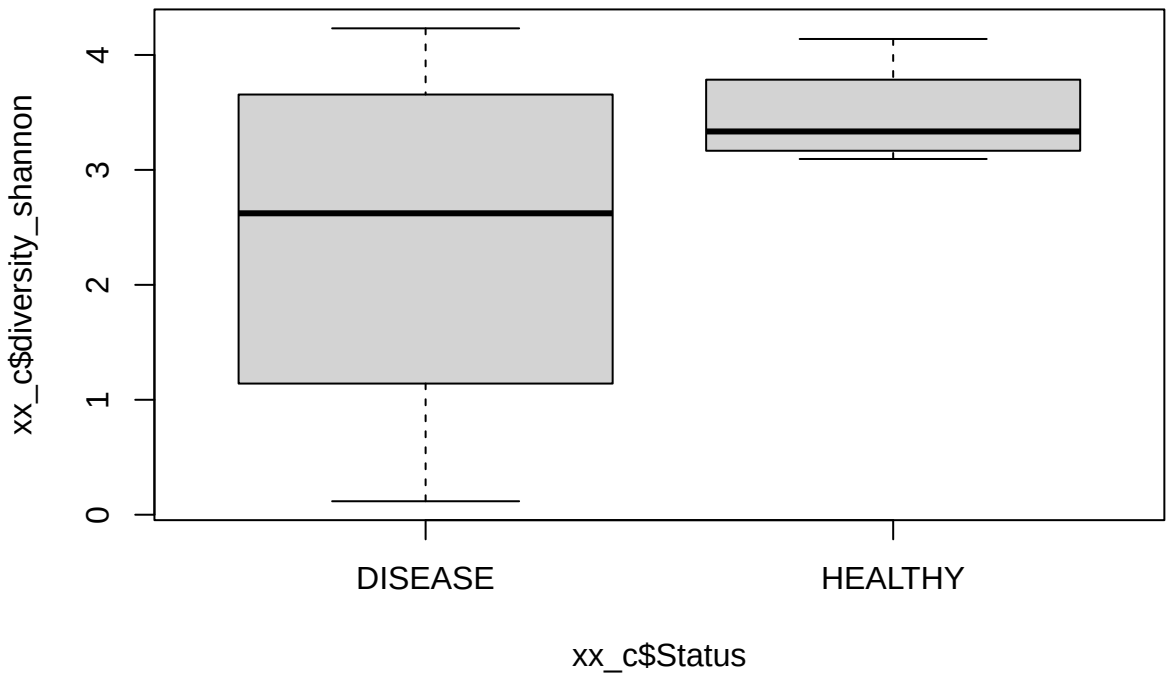
```r
### observed
suppressMessages(suppressWarnings({
    wilcox.test(xx_c$observed ~ xx_c$Status)
    boxplot(xx_c$observed ~ xx_c$Status)
}))
```



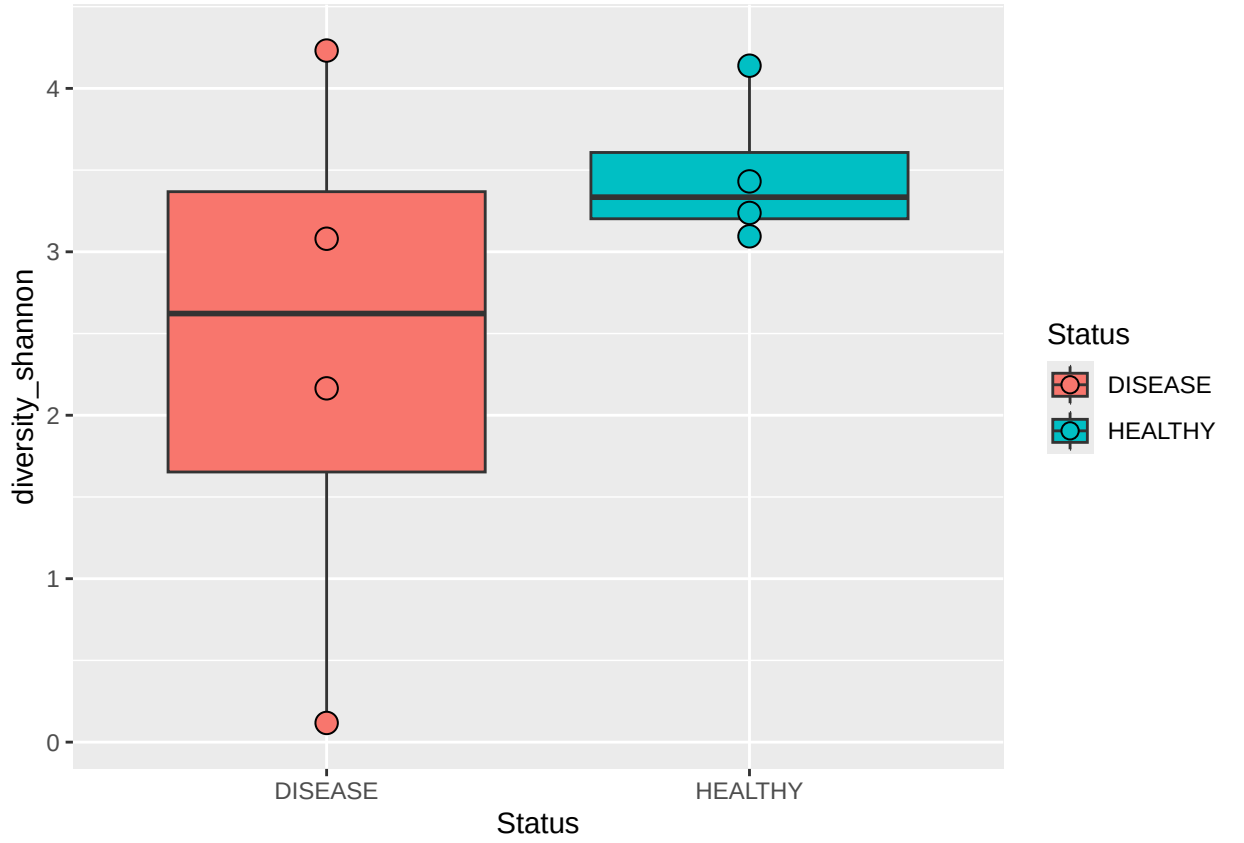

```r
ggplot(xx_c, aes(x = Status, y = observed, fill = Status)) +
    geom_boxplot() +
    geom_dotplot(
        binaxis = "y", stackdir = "center",
        binwidth = diff(range(xx$observed, na.rm = TRUE)) / 30
    )
```



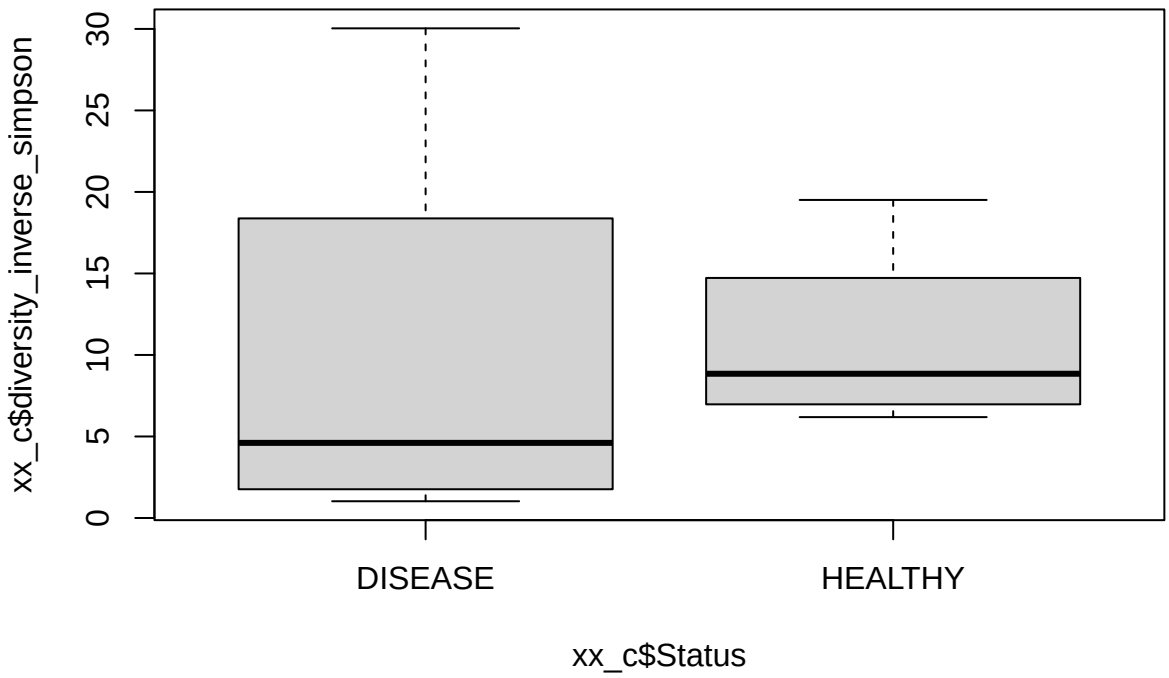

```
### diversity_shannon
suppressMessages(suppressWarnings({
    wilcox.test(xx_c$diversity_shannon ~ xx_c$Status)
    boxplot(xx_c$diversity_shannon ~ xx_c$Status)
}))
```



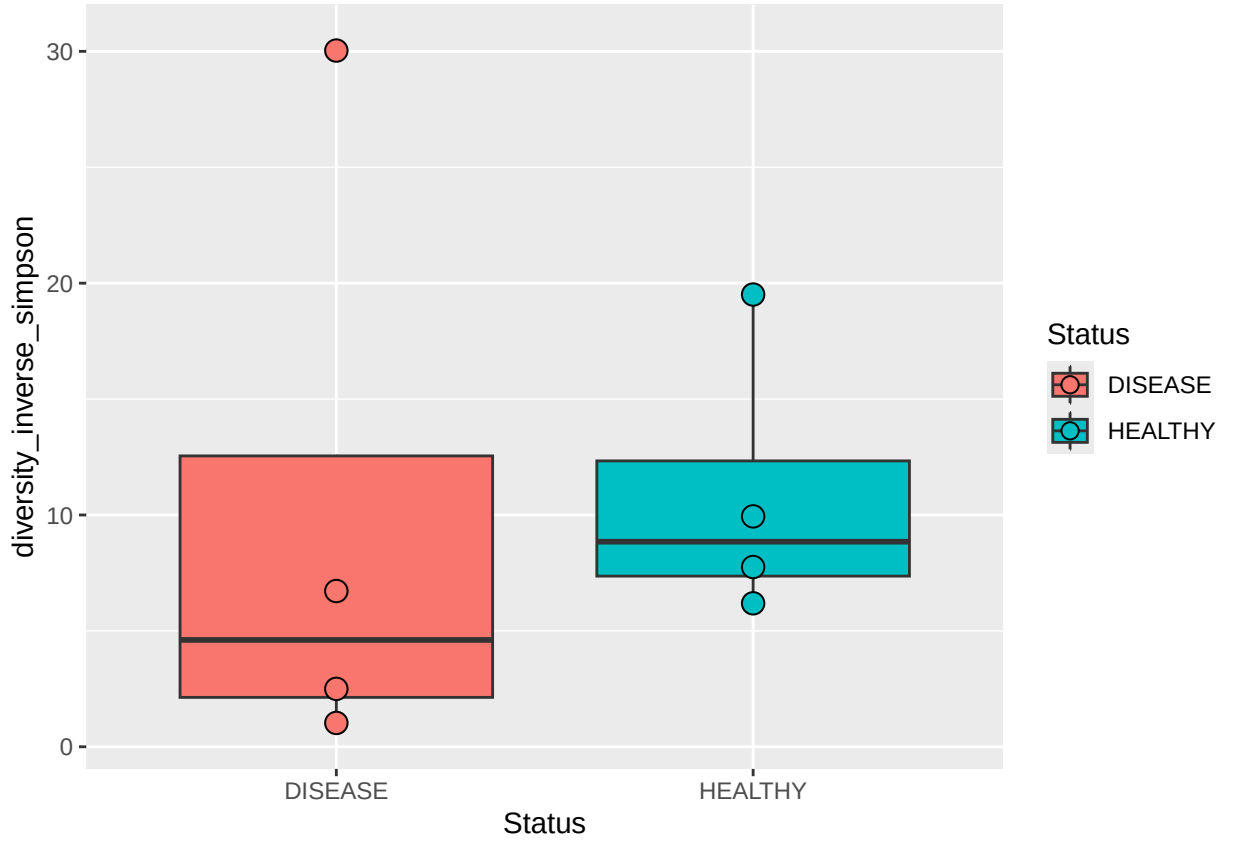

```
ggplot(xx_c, aes(x = Status, y = diversity_shannon, fill = Status)) +
    geom_boxplot() +
    geom_dotplot(
        binaxis = "y", stackdir = "center",
        binwidth = diff(range(xx$diversity_shannon, na.rm = TRUE)) / 30
    )
```




```
### diversity_inverse_simpson
suppressMessages(suppressWarnings({
    wilcox.test(xx_c$diversity_inverse_simpson ~ xx_c$Status)
    boxplot(xx_c$diversity_inverse_simpson ~ xx_c$Status)
}))
```




```
ggplot(xx_c, aes(x = Status, y = diversity_inverse_simpson, fill = Status)) +
    geom_boxplot() +
    geom_dotplot(
        binaxis = "y", stackdir = "center",
        binwidth = diff(range(xx$diversity_inverse_simpson, na.rm = TRUE)) / 30
    )
```



Questa analisi seleziona gli OTU presenti con un'abbondanza relativa >0,1% in almeno il 50% dei campioni, identificando così il core microbiota.

```
### Core microbiota
# Transform to compositional abundances
pseq.rel <- microbiome::transform(ps, "compositional")
```

```
# Pick the core (>0.1% relative abundance in >50% of the samples)
pseq.core <- core(pseq.rel, detection = 0.1 / 100, prevalence = 50 / 100)
pseq.core
```

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 30 taxa and 12 samples ]
## sample_data() Sample Data:       [ 12 samples by 2 sample variables ]
## tax_table()   Taxonomy Table:    [ 30 taxa by 7 taxonomic ranks ]
```

```
# Core with compositionals:
prevalences <- seq(.05, 1, .05)
```

```
detections <- round(10^seq(log10(5e-3), log10(.2), length = 10), 3)
```

Ora vado a far plottare questa struttura dove mi isola gli OTU che a seconda del threshold di abbondanza relativa che mi riferisco rispetto alla prevalenza sui sample che sto studiando, quali sono quelli che rimangono. Qui c'è un bilancio, se è sufficiente un 0,05% o 0,10% e se la prevalenza è superiore 50%, vediamo come il

numero di OTU che rappresenta il core diventa più piccolo. Se aumento la prevalenza di abbondanza richiesta e vedo la distribuzione di prevalenza diminuire e trovo in pochissimo sample. Se chiedo un livello di detection meno rigido, il mio core microbiota è più popolato. Quando noi leggiamo un lavoro dobbiamo:

- Vedere "i paletti" (threshoholds/ cut-offs) che avete messo per ottenere i dati.
- Vedere come sono espressi i dati
- Appuntare tutte le scelte

```
p <- plot_core(pseq.rel,
    plot.type = "heatmap",
    prevalences = prevalences, detections = detections, min.prevalence = 0.5
) +
    xlab("Detection Threshold (Relative Abundance)") +
    theme(axis.text.x = element_text(size = 9))
print(p)
```